(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平10－105372

(43)公開日　平成10年(1998)４月24日

| (51)Int.Cl.[6] | 識別記号 | ＦＩ | |
|---|---|---|---|
| Ｇ０６Ｆ　5/00 | | Ｇ０６Ｆ　5/00 | Ｈ |
| Ｈ０３Ｍ　7/30 | | Ｈ０３Ｍ　7/30 | Ｚ |

審査請求　未請求　請求項の数５　ＦＤ　（全 11 頁）

(21)出願番号　特願平8－281410

(22)出願日　平成８年(1996)10月２日

(71)出願人　591044164
株式会社沖データ
東京都港区芝浦四丁目11番地22号

(72)発明者　藤岡　恭一
東京都港区芝浦四丁目11番地22号　株式会社沖データ内

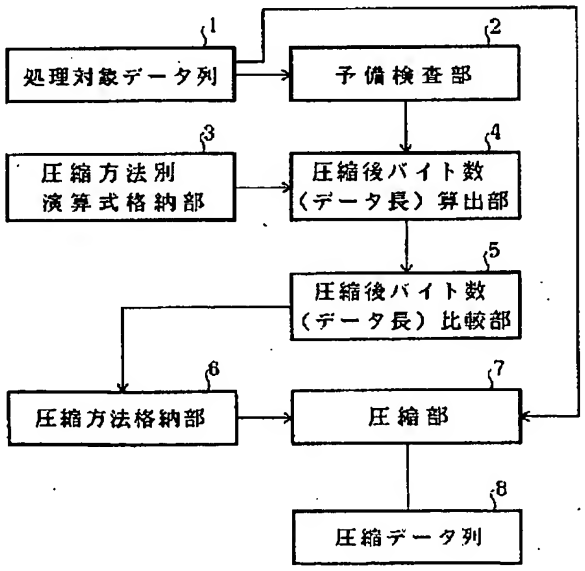
(74)代理人　弁理士　佐藤　幸男　（外１名）

(54)【発明の名称】　データ列の圧縮装置及び圧縮方法

(57)【要約】

【解決手段】　予備検査部２は処理対象データ列１を解析して、圧縮後バイト数算出のためのデータを得る。そして、複数の圧縮方式により得られる圧縮データ列のデータ長を算出した結果を圧縮後バイト数比較部５で比較する。こうしてデータ長が最短となる圧縮方法が選択され、改めて処理対象データ列がその方法で圧縮される。

【効果】　実際に全ての方法でデータを圧縮してから、その結果を比較する場合に比べて簡単な演算を用いて予測するから短時間に最適な圧縮方法が決定できる。




本発明の装置の具体例ブロック図

1

【特許請求の範囲】

【請求項１】　処理対象データ列を、予め用意された複数の圧縮方法で圧縮した場合の、圧縮データ列の長さを計算により予測する圧縮後データ長算出部と、

各圧縮方法による圧縮データ列の長さの予測結果を比較して、最適圧縮方法を選択する圧縮後データ長比較部と、

選択された圧縮方法で実際に処理対象データ列を圧縮する圧縮部とを備えたことを特徴とするデータ列の圧縮装置。

【請求項２】　処理対象データ列について、予め用意された複数の圧縮方法で圧縮した場合の、圧縮データ列の長さを、実際の圧縮処理をすることなく計算により予測して、

その計算結果を比較することにより、最適圧縮方法を選択し、

その後、前記処理対象データ列を、選択した最適圧縮方法で圧縮処理することを特徴とするデータ列の圧縮方法。

【請求項３】　請求項２において、

ＲＬＥ圧縮方法による圧縮データ長の予測値は、

処理データ列を検査して、ライン長と、繰り返し部分の合計データ長と、繰り返し部分のブロック数とを求めて、繰り返し部分のブロック数と繰り返し部分を表現するための係数長の積と、繰り返さない部分の合計データ長と繰り返さない部分を表現するための係数長の積と、繰り返し部分の合計データ長が一定値以上の場合の係数長の補正分とを加算して求めることを特徴とするデータ列の圧縮方法。

【請求項４】　請求項２において、

ＴＩＦＦ圧縮方法による圧縮データ長の予測値は、

処理データ列を検査して、ライン長と、繰り返し部分の合計データ長と、繰り返し部分のブロック数とを求めて、繰り返し部分のブロック数と繰り返し部分を表現するための係数長の積と、繰り返さない部分の合計データ長と、繰り返さない部分のブロック数と、繰り返し部分の合計データ長が一定値以上の場合の係数長の補正分と、繰り返さない部分の合計データ長が一定値以上の場合の係数長の補正分とを加算して求めることを特徴とするデータ列の圧縮方法。

【請求項５】　請求項２において、

Ｄｅｌｔａ－Ｌｏｗ圧縮方法による圧縮データ長の予測値は、

処理データ列を検査して、ライン長と、前列と異なる部分のブロック数と、前列と異なる部分の合計データ長を求めて、前列と異なる部分のブロック数と、前列と異なる部分の合計データ長と、前列と異なる部分の合計データ長が一定値以上の場合の係数長の補正分とを加算して求めることを特徴とするデータ列の圧縮方法。

【発明の詳細な説明】

2

【０００１】

【発明の属する技術分野】本発明は、イメージデータのように所定量ずつ入力するデータ列を、その中に含まれる同一データの繰り返しに着目して圧縮する場合に複数種類の圧縮方法を選択しながら最善の圧縮を実行するデータ列圧縮装置及び圧縮方法に関する。

【０００２】

【従来の技術】例えば、ホストコンピュータにおいて任意のイメージデータを生成し、これをプリンタへ転送して印刷する場合に、転送すべきデータ量を少なくしてより高速にデータを転送するためにイメージデータの圧縮処理が行われる。また、このような圧縮処理はプリンタ内にホストコンピュータから受信し印刷する直前のデータを一時保存するためのメモリ容量を削減する効果もある。こうしたイメージデータの圧縮を行う場合には、ホストコンピュータに設けられたプリンタドライバがイメージデータを圧縮処理しインタフェースを介してそのイメージデータをプリンタ側に転送する。プリンタはこうして圧縮されたイメージデータを印刷する直前に伸長してから印字ヘッドに供給する。なお、印刷用のイメージデータを圧縮する方法には数種類の方法があり、ラスタライン毎にどの圧縮方法が最適かを調べた上でいずれかの圧縮方法を選択する。

【０００３】

【発明が解決しようとする課題】ところで、上記のような従来のデータの圧縮装置や圧縮方法には次のような解決すべき課題があった。複数の圧縮方法が存在する場合に、圧縮処理の対象となるデータの内容によって、どの圧縮方法を採用した場合最もデータ長が短くなるかを調べるために、従来実際に全ての圧縮方法でイメージデータを圧縮し、その結果を比較するようにしていた。

【０００４】しかしながら、こういった従来の圧縮方法では、実際に処理対象データを圧縮処理し、何種類かの圧縮方法があれば各圧縮方法により圧縮されたデータをいったん記憶しておくためのメモリが必要となり、最適な圧縮方法を選択するために時間がかかるという問題があった。また、イメージデータを圧縮するために時間がかかれば、ホストコンピュータにおいて印刷命令を発してから実際にプリンタが印刷を開始するまでの時間が長くなるという問題もあった。

【０００５】

【課題を解決するための手段】本発明は以上の点を解決するため次の構成を採用する。

〈構成１〉処理対象データ列を、予め用意された複数の圧縮方法で圧縮した場合の、圧縮データ列の長さを計算により予測する圧縮後データ長算出部と、各圧縮方法による圧縮データ列の長さの予測結果を比較して、最適圧縮方法を選択する圧縮後データ長比較部と、選択された圧縮方法で実際に処理対象データ列を圧縮する圧縮部とを備えたことを特徴とするデータ列の圧縮装置。

【０００６】〈構成２〉処理対象データ列について、予め用意された複数の圧縮方法で圧縮した場合の、圧縮データ列の長さを、実際の圧縮処理をすることなく計算により予測して、その計算結果を比較することにより、最適圧縮方法を選択し、その後、上記処理対象データ列を、選択した最適圧縮方法で圧縮処理することを特徴とするデータ列の圧縮方法。

【０００７】〈構成３〉構成２において、ＲＬＥ圧縮方法による圧縮データ長の予測値は、処理データ列を検査して、ライン長と、繰り返し部分の合計データ長と、繰り返し部分のブロック数とを求めて、繰り返し部分のブロック数と繰り返し部分を表現するための係数長の積と、繰り返さない部分の合計データ長と繰り返さない部分を表現するための係数長の積と、繰り返し部分の合計データ長が一定値以上の場合の係数長の補正分とを加算して求めることを特徴とするデータ列の圧縮方法。

【０００８】〈構成４〉構成２において、ＴＩＦＦ圧縮方法による圧縮データ長の予測値は、処理データ列を検査して、ライン長と、繰り返し部分の合計データ長と、繰り返し部分のブロック数とを求めて、繰り返し部分のブロック数と繰り返し部分を表現するための係数長の積と、繰り返さない部分の合計データ長と、繰り返さない部分のブロック数と、繰り返し部分の合計データ長が一定値以上の場合の係数長の補正分と、繰り返さない部分の合計データ長が一定値以上の場合の係数長の補正分とを加算して求めることを特徴とするデータ列の圧縮方法。

【０００９】〈構成５〉構成２において、Ｄｅｌｔａ－Ｌｏｗ圧縮方法による圧縮データ長の予測値は、処理データ列を検査して、ライン長と、前列と異なる部分のブロック数と、前列と異なる部分の合計データ長を求めて、前列と異なる部分のブロック数と、前列と異なる部分の合計データ長と、前列と異なる部分の合計データ長が一定値以上の場合の係数長の補正分とを加算して求めることを特徴とするデータ列の圧縮方法。

【００１０】

【発明の実施の形態】以下、本発明の実施の形態を具体例を用いて説明する。

〈具体例１〉図１は、本発明の装置の具体例を示すブロック図である。この装置は、処理対象データ列１を受け入れて、実際に圧縮処理をする前に所定の検査を行う予備検査部２と、その出力を用いた圧縮後のデータ長予測演算のための式を格納する圧縮方法別演算式格納部３、圧縮後バイト数算出部４、圧縮後バイト数比較部５及び圧縮方法格納部６と、選択された最適の圧縮方法を用いて処理対象データ列１を圧縮する圧縮部７とから構成されている。この圧縮部７の出力する圧縮データ列８が例えばプリンタに送られて印刷される。

【００１１】なお、処理対象データ列１は、この例ではプリンタ用のイメージデータとする。もちろん、本発明



はプリンタ用のイメージデータの他、ファクシミリ装置、あるいはイメージデータの送受信やメモリ格納等の際に、圧縮処理することが適する各種のデータに利用できる。

【００１２】圧縮方法別演算式格納部３は、メモリから構成され、処理対象データ列１を圧縮するための各種の方法が用意されている場合に、それらの方法を用いてデータを圧縮した場合に、そのデータ長がいかなる値になるかを簡単な計算によって予測するための式が、圧縮方法に対応させて格納してある。予備検査部２は、このような演算に必要なパラメータ等を処理対象データ列１を検査することによって得る部分である。圧縮後バイト数算出部４は、圧縮方法別演算式格納部３を参照しながら圧縮方法毎に所定の演算を行って圧縮後バイト数を計算し予測する部分である。圧縮後バイト数比較部５は、圧縮方法毎の圧縮後バイト数予測値を比較し、圧縮データ列が最短となる圧縮方法を選択する部分である。圧縮方法格納部６は、圧縮後バイト数比較部５によって選択された圧縮方法を圧縮部７に供給する部分である。圧縮部７は、圧縮方法格納部６によって指定された圧縮方法を用いて処理対象データ列１を圧縮して圧縮データ列８を得る部分である。

【００１３】以上のように、本発明の装置は、処理対象データ列１を予め用意された圧縮方法で圧縮した場合に、そのデータ長がどのような値になるかを演算によって予測する。この装置の動作を説明するために、具体的な圧縮方法の例を説明する。図２は、ＲＬＥ圧縮方法の例説明図である。これから説明する例は、処理対象データ列が１バイト単位のデータにより構成され、１バイトずつ処理データ列をその先頭から検査し、同一データが連続している場合に、所定の圧縮を行う。なお、このように１バイトを単位として圧縮を行うようにしているが、一般には１ビットを単位としてもよいし、また１ワードを単位としてもよいし、圧縮のための繰り返し単位は自由に選定できる。

【００１４】図２に示す例では、例えば内容が“ＡＡ”というデータが４バイト繰り返されている。その次に、“ＯＦ”というデータと“ＦＯ”というデータが続いている。このデータをＲＬＥ圧縮方法により圧縮すると、繰り返し回数と繰り返されるイメージデータの組合せを並べて表現される。即ち、ここでは“ＡＡ”というデータが４回繰り返されているため、４に相当する２値データ“０３”を付加して、最初の圧縮データは“０３ＡＡ”となる。また、次の“ＯＦ”というデータは１回限りであるから、その圧縮後データは“００ＯＦ”となる。それに続く“ＦＯ”というデータは“００ＦＯ”というデータになる。

【００１５】図３には、ＴＩＦＦ圧縮方法の例説明図を示す。この図に示した処理対象データは、図２に示した圧縮前の処理対象データと同一の内容である。ＴＩＦＦ

5

圧縮方法においては、繰り返し数の負の値とイメージデータの組によって繰り返されるデータを表現する。また、繰り返されるデータの後に続く繰り返されないデータ列については繰り返さないバイト数－１と繰り返さないイメージデータの列によって表現される。なお、繰り返し数の負の値としたのは、繰り返されるデータの場合に、その繰り返し回数を負の値で表現し、繰り返さないイメージデータに対してはそのイメージデータの数を正で表すようにして、各データの区別ができるようにしたためである。なお、更に具体的な説明は具体例３で行う。

【００１６】図４には、Ｄｅｌｔａ－Ｒｏｗ圧縮方法の例説明図を示す。この方法は、図２や図３を用いて説明した方法と異なり、直前のラスタライン（前列）のデータとこれから圧縮をしようとする処理対象ラスタラインのデータとを比較し、同一の部分と異なる部分を求める。この圧縮方法は、特に同一内容のラスタラインが繰り返し入力するような場合にそのデータを効率よく圧縮できる。

【００１７】この処理対象データの内容は図２や図３に示したものと同一である。そして、直前のラスタラインのデータは、図の前列部分に示すように、２バイト目と５バイト目及び６バイト目が処理対象データと異なっている。こうした場合に、この圧縮方法では、異なる部分のデータ列を、コントロールバイトと、置き換えるためのイメージデータの組合せで表す。これを２値の８ビットデータで表現した場合に、上位３ビットは置き換えるバイト数－１で表す。また、下位５ビットは現在位置からのオフセットを表す。こうして得られた８ビットの２値データを１６進法のデータで表現すると、この図に示すようになる。即ち、１バイトだけ置き換える場合には、コントロールバイトは“０１”となり、置き換えるべきイメージデータ“ＡＡ”がこれに続く。また、２バイトのデータが置き換えられるとき、その現在位置からのオフセットを加味してコントロールバイトは“２２”となり、置き換えられるべきイメージデータは“ＯＦ”と“ＦＯ”となる。

【００１８】以上の圧縮方法はいずれもよく知られたプリンタ用の圧縮方法で、これらの中から最適の圧縮方法をラスタライン毎に選択していく。この場合に、本発明は次のような方法によってその圧縮方法選択を行う。

【００１９】図５には、（ａ）に本発明による圧縮方法を示し、（ｂ）に比較例による圧縮方法を示した。図において、図１に示した本発明の装置による方法では、まずステップＳ１～ステップＳ３で、予め用意されたｎ種類の圧縮方法による圧縮後のバイト数を所定の演算式を用いて算出する。そして、ステップＳ４において、どの圧縮方法を用いれば圧縮後のデータ列が最短になるかを調べる。こうして選択された最適の圧縮方法で、実際に処理対象データが圧縮される（ステップＳ４）。一方、

6

従来行われた比較例による圧縮方法では、ステップＳ１～ステップＳ３において、各圧縮方法により実際に処理対象のデータを圧縮する。そして、圧縮後のデータをメモリに格納する一方、ステップＳ４において、圧縮後のバイト数を比較する。こうして、最適の圧縮方法が選択され、メモリに格納されたいずれかのデータが出力される。

【００２０】このように、本発明において、実際にデータを圧縮することなく圧縮後のデータ長を計算して予測するようにすれば、実際に圧縮を行う場合に比べて簡単な情報を処理対象データから抽出し、しかも計算の過程では圧縮後のデータを一時保存するほど大きなメモリスペースを必要としない。従って、全体として高速に短時間に演算と比較処理ができ、小容量のバッファメモリで処理が可能となる。

【００２１】図６は、実際に図１に示した予備検査部２が取得するデータの説明図を示す。（ａ）は、ＲＬＥ圧縮及びＴＩＦＦ圧縮の際の圧縮前のデータから圧縮後バイト数を計算するために得るデータを示す。図に示すように、圧縮前のデータの長さがＬである場合に、繰り返されないデータと繰り返されるデータとが交互に存在する。このとき、繰り返されないデータの数ｍ１，ｍ２，…ｍｘ+１ と繰り返しされるデータの数ｎ１，ｎ２，…ｎｘ が存在する。

【００２２】図の（ｂ）は、Ｄｅｌｔａ－Ｒｏｗ圧縮の圧縮前のデータから取り出すべき結果の説明図である。図に示すように、前ラインと同一データはｍ１，ｍ２，…ｍｘ+１ 、異なるデータはｎ１，ｎ２，…ｎｘ 、データ長はＬといった値を求め、演算処理を行うことによってどの圧縮方法が最適かを比較できる。なお、図２～４に示した圧縮方法がなされる場合に、図６に示したような情報を得ることによって、各種の計算方法で実際に圧縮を行うことなくデータ長が演算可能であるが、以下の具体例において、それぞれ簡単に高速に演算が可能な演算方法を紹介する。

【００２３】〈具体例２〉図７に、ＲＬＥ圧縮後バイト数計算式説明図を示す。既に、図６（ａ）を用いて説明したように、処理対象データ列において、繰り返しのない部分のバイト数がｍ１，ｍ２，…ｍｘ+１ である場合に、これらの合計をＭとする（図７（ａ））。また、繰り返しがされる部分のバイト数をｎ１，ｎ２，…ｎｘ とした場合に、これらの合計をＮとする（図７（ｂ））。こうすれば、図７（ｃ）に示すように、ラスタライン全体のバイト数Ｌは、Ｍ＋Ｎによって求めることができる。

【００２４】ここで、圧縮後のバイト数を計算する式を図７（ｄ）に示す。まず、ｎ１個のデータが繰り返された場合には、基本的には１バイトの係数によって繰り返し数を表現し、もう１バイトによって繰り返されるデータを表現する。しかしながら、繰り返し数が２５７回以

上になると、１バイトではその回数を表現することはできない。即ち、２５７バイト以上繰り返される場合には、繰り返し数と繰り返されるデータの組をつなぎ合わせて表現する。そこで、何組つなぎ合わせるかを計算する式が図に示すＩＮＴ（・・）という部分である。これは、繰り返し部分を表現するための係数の補正分である。ここでは、繰り返し数ｎｌが２５６以下の場合には“１”となり、２５７以上５１２以下の場合には“２”となる。５１２を越えた場合にも順にこういった要領でＩＮＴ（・・）の部分が表現される。他の繰り返し部分についても全く同様である。なお、繰り返さない部分については１バイトの“００００”という値と繰り返さないデータとの組合せによって表現されることから、図に示すようにＬ－Ｎを２倍したものとなる。

【００２５】この図の（ｄ）に示すように、ｎｌ～ｎｘ及びＬとＮとを求めることによって圧縮後のバイト数Ｚを正確に計算することができる。なお、更に演算処理を容易にするために、図の（ｅ）には（ｄ）の式を簡略化したものを示す。この図によれば、最初のＩＮＴ（Ｎ／２５６）×２という項は（ｄ）に示した繰り返し部分の表現に使う式をくくったもので、繰り返し部分の合計バイト数Ｎを用いて繰り返し部分を表現するための係数の補正分を表している。次の第２項はＸ×２で、繰り返し部分のブロック数である。２５６バイト以上繰り返し部分がなければこの項のみで繰り返し部分が表現できる。次の（Ｌ－Ｎ）×２は、（ｄ）に示した繰り返さない部分の式をそのまま使用したものである。

【００２６】なお、第１項目の式は繰り返し部分の合計バイト数Ｎを用いて一括して係数の補正分を表したため若干の誤差が生じる。しかしながら、最小限このバイト数に圧縮されるという結果が得られることから、他の圧縮方法との比較が可能になる。こうすれば、図に示すように繰り返し部分の合計バイト数即ち合計データ長であるＮと、繰り返し部分のブロック数Ｘと、ラスタライン全体のバイト数Ｌによって簡単な式で圧縮後のバイト数を計算することができる。Ｎ，Ｘ，Ｌを求めるための検査は高速かつ容易である。従って、従来のように実際に圧縮処理を行う場合に比べて極めて高速に演算処理ができる。しかも、このＮ，Ｘ，Ｌの値は他の圧縮方法による圧縮後のバイト数を予測するためにも利用できるため、データの検査は一部省略できるという効果もある。なお、Ｎ，Ｘ，Ｌを求める代わりにＭ，Ｘ，Ｌを求めるようにしてもよいし、これらの値を別の方法により間接的に求めてもよい。

【００２７】図８に、ＲＬＥ圧縮後のバイト数算出法フローチャートを示す。上記の計算は具体的にはこの図に示すような手順によって行われる。なお、この処理を行うために、データ読込みポインタＳＲＣとデータ数カウンタＣＯＵＮＴと、圧縮データ数カウンタＺ、繰り返し部分バイト数カウンタＮ、繰り返し部分ブロック数カウンタＸ、繰り返し部分中フラグＦとが用意される。このように、この計算のためには小容量の簡単なバッファのみが使用される。



【００２８】なお、データ読込みポインタＳＲＣは、処理対象データ列の何番目のデータを読み込んだかを示すポインタである。また、データ数カウンタＣＯＵＮＴは最初にＬをセットオフしておき、１ライン分全てのデータについて処理が終了したかを判断するためのパラメータである。Ｚ，Ｎ，Ｘは、既に説明したとおりのもので、繰り返し部分中フラグＦは、ポインタがデータの繰り返し部分にくると“１”になり、繰り返し部分外では“０”になるパラメータである。

【００２９】図８のステップＳ１において、まず繰り返し部分バイト数カウンタＮと繰り返し部分ブロック数カウンタＸとをゼロクリアし、フラグＦも“０”にリセットする。そして、ステップＳ２において、まず隣接データ即ち直前のデータと次のデータとの比較を行い、これらが等しいかどうかを判断する。そして、等しければステップＳ３に進み、フラグＦが“０”かどうかを判断する。フラグＦが“０”でない場合にはステップＳ４に移り、フラグＦを“１”にセットして、ステップＳ５で繰り返し部分ブロック数カウンタＸをインクリメントする。即ち、初めて繰り返しブロックが検出されると、そのブロック数Ｘがカウントされる。更に、ステップＳ６において、繰り返しバイト数がインクリメントされる。

【００３０】既に繰り返し部分の先頭が検出され、繰り返し数のみをカウントする場合には、ステップＳ３からステップＳ６にジャンプして繰り返し部分バイト数のみがインクリメントされる。ステップＳ７ではポインタを“１”だけインクリメントし、ステップＳ８ではカウンタをデクリメントする。ステップＳ９で、カウンタが“１”かどうかを判断する。こうして、全ての処理対象データについて処理を終了したかどうかを判断する。

【００３１】終了していなければ、再びステップＳ２に移る。直前のデータと次のデータとが異なる場合には、ステップＳ２からステップＳ１３に進み、フラグＦが“０”にセットされる。こうして、全てのデータについて、繰り返し部分のブロック数Ｘと繰り返し部分のバイト数Ｎとを検出する。そして、これらの処理を終了するとステップＳ１０に移り、最終データとその直前のデータとが等しかったかどうかを判断する。等しかった場合には、最終データの分について、繰り返し部分バイト数Ｎをインクリメントする。等しくない場合には、ステップＳ１１をパスし、ステップＳ１２に進む。こうしてステップＳ１２において、上記図７に示した演算処理を実行する。

【００３２】〈具体例２の効果〉以上の演算方法により、処理対象データから簡単な情報を得ることによって、正確な圧縮後のデータサイズを予測することができる。

【００３３】〈具体例３〉図９は、ＴＩＦＦ圧縮後バイト数計算式の説明図である。この図の（ａ）、（ｂ）、（ｃ）は、既に図７を用いて説明した繰り返さない部分のバイト数Ｍ、繰り返し部分のバイト数Ｎ及びラスタライン全体のバイト数Ｌを示す。ここで、この具体例では、（ｄ）に示すようにして、圧縮後のバイト数を計算する。図の最初の項のＩＮＴ（・・）は、１２８バイト以上同一データが繰り返された場合の繰り返し数とデータとの組の数を表す。図７の例では２５６であったのに、この例では１２８となったのは次の理由による。

【００３４】ＴＩＦＦ圧縮処理においては、８バイトのデータによって表現できる２５６の数字を前半の１２８と後半の１２８とに分ける。そして、前半を０～１２７まで表現し、後半を－１～－１２７で表現するように割り当てる。これによって、負の係数で繰り返し数を表し、正の係数で繰り返さないデータのバイト数を表す。その結果、最大１２８バイトの連続しか表現できないためこの図９（ｄ）に示すような係数長の補正が行われる。その他の原理は図７を用いて説明したものと同一である。なお、繰り返さない部分については、繰り返さない部分のバイト数と繰り返さない部分のデータの数ｍ１とによって表現されるから、第３項目、第４項目のような結果となる。（ｅ）には、圧縮後のバイト数を簡略化した計算式を示した。

【００３５】図９の（ｅ）式に示すように、簡略化した式は５項目の和により圧縮後のバイト数Ｚを求める。最初の項ＩＮＴ（Ｎ／１２８）×２は具体例２の図７に示したケースと全く同様にして、図９（ｄ）の繰り返し部分の係数をまとめたものである。従って、この式は繰り返し部分のバイト数Ｎのみを用いて求めることができる。これによって、繰り返し部分の表現に用いる係数の係数長の補正分が表される。次の第２項、Ｘ×２は繰り返し部分のブロック数を２倍したもので、繰り返し部分が１２８バイト以下のものばかりであればこの項のみで表現できる。次のＩＮＴ（Ｌ－Ｎ／１２８）は、繰り返さない部分を表現する係数の成分である。次のＸ＋１は繰り返さない部分のブロック数である。繰り返さない部分の全てのブロックが１２８バイト以下の場合にはこの項だけで係数が表現できる。Ｌ－Ｎは繰り返さない部分の合計バイト数である。この計算も具体例２で求めた繰り返し部分のバイト数Ｎ、ラスタライン全体のバイト数Ｌ、繰り返し部分のブロック数Ｘの３種のデータによって簡単に演算できる。その他の係数や算出用データの取扱いは具体例２を用いて説明したものと同様である。

【００３６】図１０には、ＴＩＦＦ圧縮後のバイト数算出方法フローチャートを示す。この図に示す処理は、具体的には図８に示す処理とほぼ同様となる。即ち、両者を比較してわかるように、その差はステップＳ１２における集計部分のみである。このように異なる圧縮方法を用いた場合の圧縮後のバイト数をほぼ共通のプログラムを用いて算出することにより、極めて高速に圧縮後のバイト数予測が可能となる。従って、従来のように、それぞれ別々の異なる圧縮方法を用いて具体的に圧縮処理した後バイト数をカウントする方法に比べて著しく圧縮結果の比較演算を高速化することができる。

【００３７】〈具体例４〉図１１に、Ｄｅｌｔａ－Ｒｏｗ圧縮後バイト数計算式説明図を示す。この例においては、（ａ）に示すように、前列と同一部分のバイト数Ｍと、（ｂ）に示すように、前列と異なる部分のバイト数Ｎと、（ｃ）に示すラスタライン全体のバイト数Ｌとを求める。前列と同一部分の合計バイト数Ｍは、既に図６を用いて説明したとおり、ｍ１ ＋ｍ２ ＋ｍｘ＋１ となる。また、前列と異なる部分の合計バイト数Ｎはｎ１ ＋ｎ２ ＋…ｎｘ となる。ここで、圧縮後のバイト数Ｚは、（ｄ）に示すとおり演算される。（ｄ）の式の第１項は異なる部分を示し、図４に示したように置換バイト数を３ビットで表すため、８個以上の場合には、このデータが２組以上繰り返される。そこで、ＩＮＴ（・・）の部分がこの図に示すように補正用に演算される。また、同一部分については、第３番目、第４番目の項及びｎ１ ～ｎｘ までのバイト数のデータが付け加えられる。（ｅ）に示したものは、これらを簡略化した演算式である。これも具体例２や具体例３と同様に高速演算のために簡略化を行い、圧縮後のバイト数Ｚの最大値を求めるようにしている。

【００３８】図１１の（ｅ）において、第１項目は異なる部分の長さを表現するための係数長の補正分である。Ｘは前列と異なる部分のブロック数で、第３項目はオフセット位置を示すための係数長補正分である。これに前列と異なる部分のブロック数Ｘとそのバイト数Ｎを加算する。このように（ｄ）式を簡略化し、最小限このバイト数に圧縮されるという結果を得て他の圧縮方式と比較する。なお、この例では、Ｍ，Ｎは具体例２，具体例３と別の値であって、別の方法によって検査し求めることになる。しかしながら、やはりＭ，Ｎ，Ｌといった簡単な数値を用いて演算処理を行うため高速で圧縮後のバイト数算出が可能となる。

【００３９】図１２には、Ｄｅｌｔａ－Ｒｏｗ圧縮後のバイト数算出法フローチャートを示す。ここでは、新たに前列データポインタＳＥＥＤをセットし、前列と圧縮対象データ列とを比較しながら処理を行う。ステップＳ２，Ｓ３，Ｓ４，Ｓ５，Ｓ６及びステップＳ１２の処理は、具体例２、具体例３と同様である。また、ステップＳ７では、前列と処理対象データ列とのポインタを同期させるために、前列データポインタＳＥＥＤをインクリメントしている。ステップＳ８，Ｓ９，Ｓ１０は、これまでの具体例の処理と同様である。こうして、ステップＳ１１において、比較処理に必要な圧縮後のバイト数Ｚを得る。

【００４０】〈具体例４の効果〉この具体例において

11

も、実際に処理対象データ列を圧縮処理することなく圧縮処理後のデータ長を高速演算処理することが可能になる。

【図面の簡単な説明】

【図１】本発明の装置の具体例を示すブロック図である。

【図２】ＲＬＥ圧縮方法の例説明図である。

【図３】ＴＩＦＦ圧縮方法の例説明図である。

【図４】Ｄｅｌｔａ－Ｒｏｗ圧縮方法の例説明図である。

【図５】本発明による圧縮方法と比較例による圧縮方法のフローチャートである。

【図６】各圧縮方法の圧縮前のデータから得る情報の説明図である。

【図７】ＲＬＥ圧縮後バイト数計算式説明図である。

【図８】ＲＬＥ圧縮後のバイト数算出法フローチャートである。

12

【図９】ＴＩＦＦ圧縮後バイト数計算式説明図である。

【図１０】ＴＩＦＦ圧縮後のバイト数算出法フローチャートである。

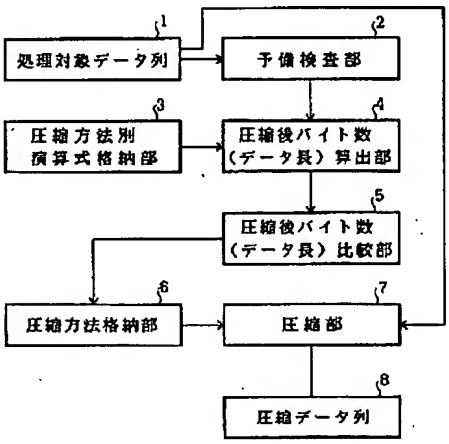
【図１１】Ｄｅｌｔａ－Ｒｏｗ圧縮後バイト数計算式説明図である。

【図１２】Ｄｅｌｔａ－Ｒｏｗ圧縮後のバイト数算出法フローチャートである。

【符号の説明】

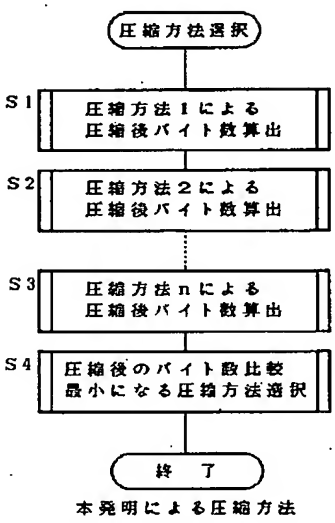
1　処理対象データ列
2　予備検査部
3　圧縮方法別演算式格納部
4　圧縮後バイト数算出部
5　圧縮後バイト数比較部
6　圧縮方法格納部
7　圧縮部
8　圧縮データ列

【図１】

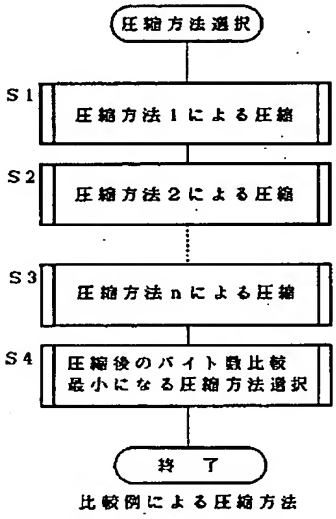



本発明の装置の具体例ブロック図

【図５】

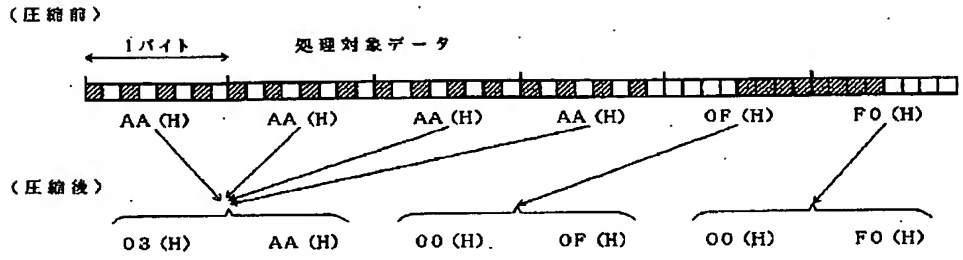



本発明による圧縮方法
（ａ）



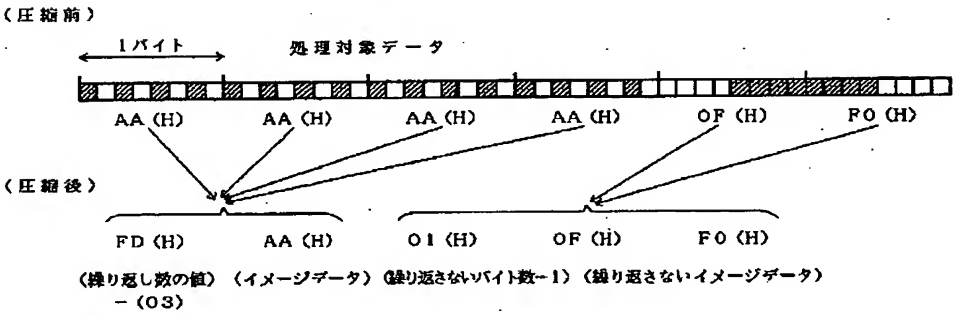

比較例による圧縮方法
（ｂ）

【図２】



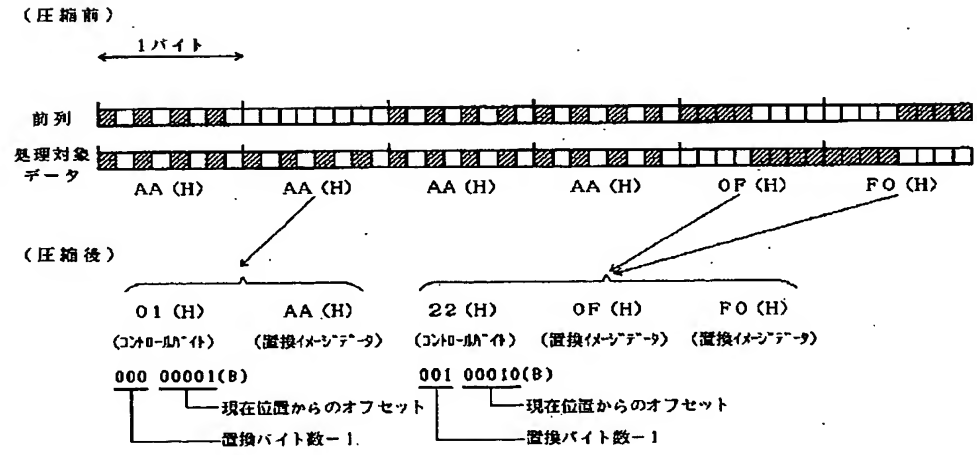

ＲＬＥ圧縮方法の例説明図

【図３】



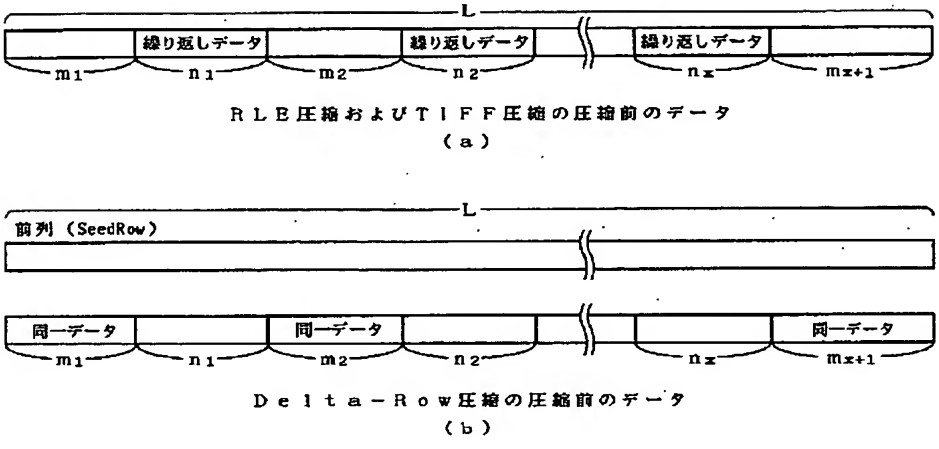

ＴＩＦＦ圧縮方法の例説明図

【図４】




Ｄｅｌｔａ－Ｒｏｗ圧縮方法の例説明図

【図６】




【図７】

（ａ）繰り返さない部分のバイト数　$M = m_1 + m_2 + \cdots + m_{x+1}$

（ｂ）繰り返す部分のバイト数　$N = n_1 + n_2 + \cdots + n_x$

（ｃ）ラスタライン全体のバイト数　$L = M + N$

（ｄ）圧縮後のバイト数

$$Z = INT\left(\frac{n_1+255}{256}\right) * 2$$　$n_1$個繰り返し分

$$+ INT\left(\frac{n_2+255}{256}\right) * 2$$　$n_2$個繰り返し分

……

$$+ INT\left(\frac{n_x+255}{256}\right) * 2$$　$n_x$個繰り返し分

$+ (L-N) * 2$　繰り返さない分

（ｅ）圧縮後のバイト数（簡略式）

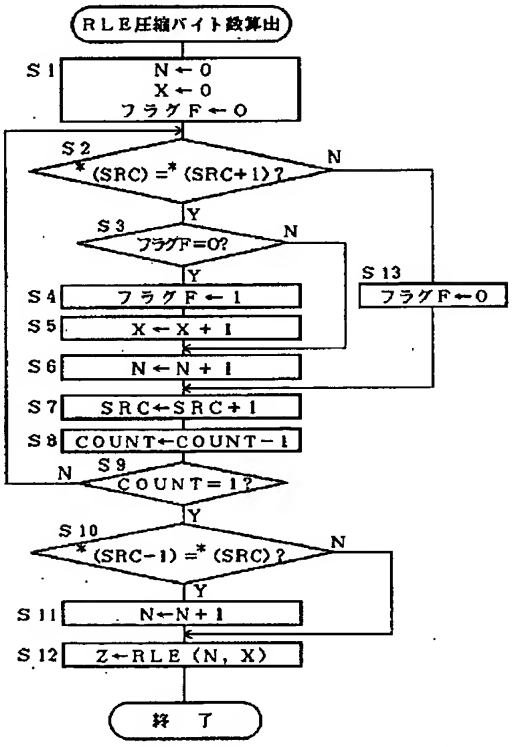
$$Z \leqq INT\left(\frac{N}{256}\right) * 2$$　２５６バイト以上の繰り返し分

$+ \quad X \quad * 2$　繰り返し分

$+ (L-N) * 2$　繰り返さない分

ＲＬＥ圧縮後バイト数計算式説明図

【図８】




ＲＬＥ圧縮後のバイト数算出法

【図９】

（ａ）繰り返さない部分のバイト数　$M=m_1+m_2+\cdots+m_{x+1}$

（ｂ）繰り返す部分のバイト数　$N=n_1+n_2+\cdots+n_x$

（ｃ）ラスタライン全体のバイト数　$L=M+N$

（ｄ）圧縮後のバイト数

$Z=INT\left(\frac{n_1+127}{128}\right)*2$　繰り返し分

……

$+INT\left(\frac{n_x+127}{128}\right)*2$　繰り返し分

$+INT\left(\frac{m_1+127}{128}\right)+m_1$　繰り返さない分

……

$+INT\left(\frac{m_{x+1}+127}{128}\right)+m_{x+1}$　繰り返さない分

（ｅ）圧縮後のバイト数（簡略式）

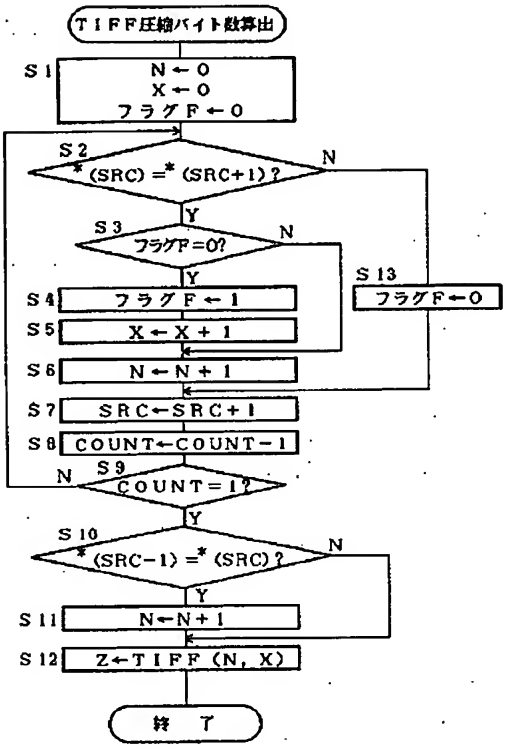
$Z\leqq INT\left(\frac{N}{128}\right)*2+x*2$　繰り返し分

$+INT\left(\frac{L-N}{128}\right)+(x+1)$　繰り返さない分

$+L-N$　繰り返さない分

ＴＩＦＦ圧縮後バイト数計算式説明図

【図１０】




ＴＩＦＦ圧縮後のバイト数算出法

【図１１】

（ａ）前列と同一部分のバイト数　$M=m_1+m_2+\cdots+m_{x+1}$

（ｂ）前列と異なる部分のバイト数　$N=n_1+n_2+\cdots+n_x$

（ｃ）ラスタライン全体のバイト数　$L=M+N$

（ｄ）圧縮後のバイト数

$Z=INT\left(\frac{n_1+7}{8}\right)$　異なる部分

……

$+INT\left(\frac{n_x+7}{8}\right)$　異なる部分

$+INT\left(\frac{(m_1-30)+254}{255}\right)$　同一の部分

……

$+INT\left(\frac{(m_x-30)+254}{255}\right)$　同一の部分

$+n_1+n_2++n_x$

（ｅ）圧縮後のバイト数（簡略式）

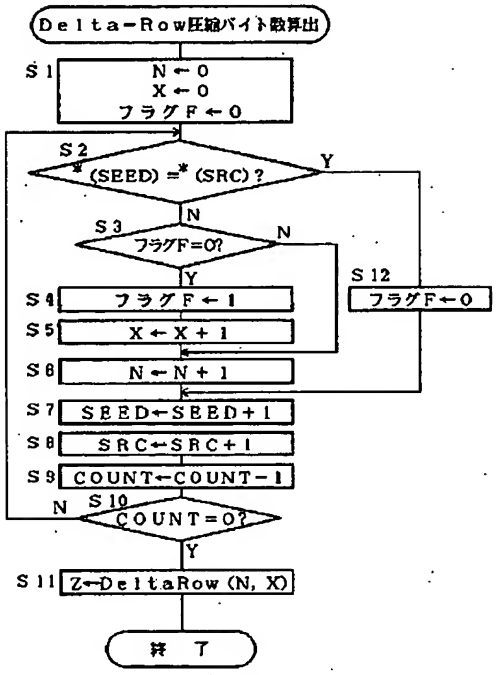
$Z\leqq INT\left(\frac{N}{8}\right)+x$

$+INT\left(\frac{M-30*x}{255}\right)+x$

$+N$

Ｄｅｌｔａ－Ｒｏｗ圧縮後バイト数計算式説明図

【図１２】




Ｄｅｌｔａ－Ｒｏｗ圧縮後のバイト数算出法